

Chapter 14

Web Database Programming Using PHP



Addison-Wesley
is an imprint of

PEARSON

Chapter 14 Outline

- A Simple PHP Example
- Overview of Basic Features of PHP
- Overview of PHP Database Programming

Web Database Programming Using PHP

- Techniques for programming dynamic features into Web
- PHP
 - Open source scripting language
 - Interpreters provided free of charge
 - Available on most computer platforms

A Simple PHP Example

- PHP
 - Open source general-purpose scripting language
 - Comes installed with the UNIX operating system

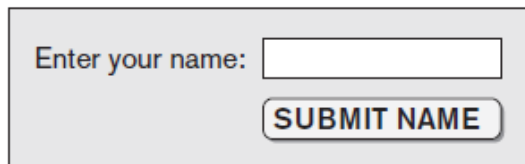
A Simple PHP Example (cont'd.)

- **DBMS**
 - **Bottom-tier database server**
- **PHP**
 - **Middle-tier Web server**
- **HTML**
 - **Client tier**

(a)

```
//Program Segment P1:
0) <?php
1) // Printing a welcome message if the user submitted their name
   // through the HTML form
2) if ($_POST['user_name']) {
3)     print("Welcome,  ") ;
4)     print($_POST['user_name']);
5) }
6) else {
7)     // Printing the form to enter the user name since no name has
       // been entered yet
8)     print <<<_HTML_
9)     <FORM method="post" action="$_SERVER['PHP_SELF']">
10)    Enter your name: <input type="text" name="user_name">
11)    <BR/>
12)    <INPUT type="submit" value="SUBMIT NAME">
13)    </FORM>
14)    _HTML_;
15) }
16) ?>
```

(b)



Enter your name:

(c)



Enter your name:

(d)



Welcome, John Smith

Figure 14.1

(a) PHP program segment for entering a greeting, (b) Initial form displayed by PHP program segment, (c) User enters name *John Smith*, (d) Form prints welcome message for *John Smith*.

A Simple PHP Example (cont'd.)

- Example Figure 14.1(a)
- PHP script stored in:
 - <http://www.myserver.com/example/greeting.php>
- `<?php`
 - PHP start tag
- `?>`
 - PHP end tag
- Comments: `//` or `* * /`

A Simple PHP Example (cont'd.)

- `$_POST`
 - **Auto-global** predefined PHP variable
 - Array that holds all the values entered through form parameters
- Arrays are dynamic
- **Long text strings**
 - Between opening `<<<_HTML_` and closing `_HTML_;`

A Simple PHP Example (cont'd.)

- **PHP variable names**
 - Start with \$ sign

Overview of Basic Features of PHP

- Illustrate features of PHP suited for creating dynamic Web pages that contain database access commands

PHP Variables, Data Types, and Programming Constructs

- **PHP variable names**
 - Start with \$ symbol
 - Can include characters, letters, and underscore character (_)
- **Main ways to express strings and text**
 - **Single-quoted strings**
 - **Double-quoted strings**
 - **Here documents**
 - **Single and double quotes**

PHP Variables, Data Types, and Programming Constructs (cont'd.)

- Period (.) symbol
 - String concatenate operator
- Single-quoted strings
 - Literal strings that contain no PHP program variables
- Double-quoted strings and here documents
 - Values from variables need to be interpolated into string

PHP Variables, Data Types, and Programming Constructs (cont'd.)

- Numeric data types
 - Integers and floating points
- Programming language constructs
 - For-loops, while-loops, and conditional if-statements
- Boolean expressions

PHP Variables, Data Types, and Programming Constructs (cont'd.)

- Comparison operators
 - == (equal), != (not equal), > (greater than), >= (greater than or equal), < (less than), and <= (less than or equal)

PHP Arrays

- Database query results
 - Two-dimensional arrays
 - First dimension representing rows of a table
 - Second dimension representing columns (attributes) within a row
- Main types of arrays:
 - **Numeric** and **associative**

PHP Arrays (cont'd.)

- Numeric array
 - Associates a numeric index with each element in the array
 - Indexes are integer numbers
 - Start at zero
 - Grow incrementally
- Associative array
 - Provides pairs of (key => value) elements

PHP Arrays (cont'd.)

Figure 14.3

Illustrating basic PHP array processing.

```
0) $teaching = array('Database' => 'Smith', 'OS' => 'Carrick',  
                    'Graphics' => 'Kam');  
1) $teaching['Graphics'] = 'Benson'; $teaching['Data Mining'] = 'Kam';  
2) sort($teaching);  
3) foreach ($teaching as $key => $value) {  
4)     print " $key : $value\n";}  
5) $courses = array('Database', 'OS', 'Graphics', 'Data Mining');  
6) $alt_row_color = array('blue', 'yellow');  
7) for ($i = 0, $num = count($courses); i < $num; $i++) {  
8)     print '<TR bgcolor="' . $alt_row_color[$i % 2] . '">';  
9)     print "<TD>Course $i is</TD><TD>$course[$i]</TD></TR>\n";  
10) }
```

PHP Arrays (cont'd.)

- Techniques for looping through arrays in PHP
- Count function
 - Returns current number of elements in array
- Sort function
 - Sorts array based on element values in it

PHP Functions

- Functions
 - Define to structure a complex program and to share common sections of code
 - Arguments passed by value
- Examples to illustrate basic PHP functions
 - Figure 14.4
 - Figure 14.5

Figure 14.4

Rewriting program segment P1 as P1' using functions.

```
//Program Segment P1':
0) function display_welcome() {
1)     print("Welcome,  ") ;
2)     print($_POST['user_name']);
3) }
4)
5) function display_empty_form(); {
6) print <<<_HTML_
7) <FORM method="post" action="$_SERVER['PHP_SELF']">
8) Enter your name: <INPUT type="text" name="user_name">
9) <BR/>
10) <INPUT type="submit" value="Submit name">
11) </FORM>
12) _HTML_;
13) }
14) if ($_POST['user_name']) {
15)     display_welcome();
16) }
17) else {
18)     display_empty_form();
19) }
```

PHP Functions (cont'd.)

Figure 14.5

Illustrating a function with arguments and return value.

```
0) function course_instructor ($course, $teaching_assignments) {
1)     if (array_key_exists($course, $teaching_assignments)) {
2)         $instructor = $teaching_assignments[$course];
3)         RETURN "$instructor is teaching $course";
4)     }
5)     else {
6)         RETURN "there is no $course course";
7)     }
8) }
9) $teaching = array('Database' => 'Smith', 'OS' => 'Carrick',
                    'Graphics' => 'Kam');
10) $teaching['Graphics'] = 'Benson'; $teaching['Data Mining'] = 'Kam';
11) $x = course_instructor('Database', $teaching);
12) print($x);
13) $x = course_instructor('Computer Architecture', $teaching);
14) print($x);
```

PHP Server Variables and Forms

- Built-in entries
 - `$_SERVER` auto-global built-in array variable
 - Provides useful information about server where the PHP interpreter is running

PHP Server Variables and Forms (cont'd.)

- Examples:

- `$_SERVER['SERVER_NAME']`
- `$_SERVER['REMOTE_ADDRESS']`
- `$_SERVER['REMOTE_HOST']`
- `$_SERVER['PATH_INFO']`
- `$_SERVER['QUERY_STRING']`
- `$_SERVER['DOCUMENT_ROOT']`

- `$_POST`

- Provides input values submitted by the user through HTML forms specified in `<INPUT>` tag

Overview of PHP Database Programming

- PEAR DB library
 - Part of PHP Extension and Application Repository (PEAR)
 - Provides functions for database access

Connecting to a Database

- Library module DB.php must be loaded
- DB library functions accessed using `DB::<function_name>`
- `DB::connect('string')`
 - Function for connecting to a database
 - Format for 'string' is: `<DBMS software>://<user account>:<password>@<database server>`

```

0) require 'DB.php';
1) $d = DB::connect('oci8://acctl:pass12@www.host.com/db1');
2) if (DB::isError($d)) { die("cannot connect - " . $d->getMessage());}
   ...
3) $q = $d->query("CREATE TABLE EMPLOYEE
4)   (Emp_id INT,
5)   Name VARCHAR(15),
6)   Job VARCHAR(10),
7)   Dno INT)" );
8) if (DB::isError($q)) { die("table creation not successful - " .
   $q->getMessage()); }
   ...
9) $d->setErrorHandler(PEAR_ERROR_DIE);
   ...
10) $eid = $d->nextID('EMPLOYEE');
11) $q = $d->query("INSERT INTO EMPLOYEE VALUES
12)   ($eid, $_POST['emp_name'], $_POST['emp_job'], $_POST['emp_dno'])" );
   ...
13) $eid = $d->nextID('EMPLOYEE');
14) $q = $d->query('INSERT INTO EMPLOYEE VALUES (?, ?, ?, ?)',
15) array($eid, $_POST['emp_name'], $_POST['emp_job'], $_POST['emp_dno']) );

```

Figure 14.6

Connecting to a database, creating a table, and inserting a record.

Connecting to a Database (cont'd.)

- Query function
 - `$d->query` takes an SQL command as its string argument
 - Sends query to database server for execution
- `$d->setErrorHandler(PEAR_ERROR_DIE)`
 - Terminate program and print default error messages if any subsequent errors occur

Collecting Data from Forms and Inserting Records

- Collect information through HTML or other types of Web forms
- Create unique record identifier for each new record inserted into the database
- PHP has a function `$d->nextID` to create a sequence of unique values for a particular table
- **Placeholders**
 - Specified by `?` symbol

Retrieval Queries from Database Tables

- `$q`
 - **Query result**
 - `$q->fetchRow()` retrieve next record in query result and control loop
- `$d=>getAll`
 - Holds all the records in a query result in a single variable called `$allresult`

```

0) require 'DB.php';
1) $d = DB::connect('oci8://acctl:pass12@www.host.com/dbname');
2) if (DB::isError($d)) { die("cannot connect - " . $d->getMessage()); }
3) $d->setErrorHandler(PEAR_ERROR_DIE);
   ...
4) $q = $d->query('SELECT Name, Dno FROM EMPLOYEE');
5) while ($r = $q->fetchRow()) {
6)     print "employee $r[0] works for department $r[1] \n" ;
7) }
   ...
8) $q = $d->query('SELECT Name FROM EMPLOYEE WHERE Job = ? AND Dno = ?',
9)     array($_POST['emp_job'], $_POST['emp_dno']) );
10) print "employees in dept $_POST['emp_dno'] whose job is
      $_POST['emp_job']: \n"
11) while ($r = $q->fetchRow()) {
12)     print "employee $r[0] \n" ;
13) }
   ...
14) $allresult = $d->getAll('SELECT Name, Job, Dno FROM EMPLOYEE');
15) foreach ($allresult as $r) {
16)     print "employee $r[0] has job $r[1] and works for department $r[2] \n" ;
17) }
   ...

```

Figure 14.7

Illustrating database retrieval queries.

Summary

- PHP scripting language
 - Very popular for Web database programming
- PHP basics for Web programming
- Data types
- Database commands include:
 - Creating tables, inserting new records, and retrieving database records